The Online Community Grid
Volunteer Grid Computing with the Web Browser

A Thesis

Presented to

The Academic Faculty

By

Daniel Menachem Miller

Georgia Institute of Technology April 2008

Abstract

The Online Community Grid (OCG) is a framework, began in 2006, for distributed and volunteer computing in Adobe Flash 9. About 97% of Internet users have Adobe Flash installed on their systems and no additional software is needed to work with the OCG. The OCG, shown as a widget on a web page, runs entirely on a web browser and collaborates with web servers via HTTP. Utilizing open source libraries such as AsWing, FastDLL, coreLib, and xmlSyndicationLib, the OCG situates on a central server and loads libraries established on external web servers and maintained by third parties. Upon review of various alternative browser-based technologies, Adobe Flash 9 powered with Actionscript 3.0 performs the fastest. Following comparisons among existing, standalone frameworks, the Online Community Grid is more portable than standalone frameworks such as BOINC and the World Community Grid. No registration is required to join the OCG, and malicious/invalid data is handled via duplication and server validation (only applicable with certain problems). Similar to BOINC, the OCG handles client CPU strain and provides the ability to store checkpoints on the client system as a SharedObject.

1 Background

Grid computing is by no means a new field, and with the advent of new systems and parallel algorithms, vast networks such as SETI@home and Folding@home are out-performing and out-pricing the world's fastest super computers. The world's fastest supercomputer is IBM's Blue Gene which, as of November 2007, has a sustained processing rate of 478.2 teraflops. Blue Gene has cost hundreds of millions of dollars to build and maintain[1]. Significantly cheaper and faster sources of computing power are grid projects such as SETI@home or Folding@home. These grid computing projects have hundreds of thousands of participants and have passed the petaflop scale[2].

The very first and oldest large scale volunteer grid computing project is the Great Internet Mersenne Prime Search (GIMPS) [3], which began in 1996. The software uses thousands of computers to search for specific large prime numbers. The project proved to be a

great success, and since the project's debut, ten record-breaking Mersenne Prime numbers have been found with GIMPS.

A few years later, in 1999, SETI@home is released. SETI@home is a grid computing project that, unlike GIMPS, focuses on a non mathematical application. Hosted at the Space Sciences Laboratory at the University of California, Berkley, SETI@home is an acronym for "Search for Extra-Terrestrial Intelligence." The 1999 SETI@home introduction article, "ET, phone SETI@home" [4] explains that the project uses thousands of computers to processes observational data from the farthest reaches of space. The project runs as a screensaver and sends data results from the processes back to a central server. While the project has yet to find signs of extra-terrestrial intelligence, SETI@home proves [5] the capacity of the distributed computing grid concept and inspired later projects and frameworks.

A 2000 article in *Science* [6] introduced Folding@home, a network of PCs dedicated to solving humanitarian problems such as climate change (by simulating weather patterns), diseases such as HIV (by simulating protein folding and misfolding), and finding disease-provoking genes (by performing genetic linkage analysis). Maintained by the chemistry department at Stanford, Folding@home is similar to SETI@home as its users run a downloaded screensaver which simulates protein structure folding. Folding@home has recently been ported to Playstation 3. The additions of Playstation 3 computing power has grown Folding@home to the most popular grid computing project in existence [7] with a sustained peak at 1224 teraflops.

The Berkeley Open Infrastructure for Network Computing (BOINC) was introduced in *New Science* at the end 2003 [8]. This framework software is built to allow users to choose between a variety of grid computing projects. Scientists can develop grid computing projects with the BOINC platform and users can then select to join those projects. The BOINC platform has led to hundreds of distributed computing projects such as POEM@home and ABC@home.

The first version of BOINC was a big step forward for grid computing; however, while these applications and systems are organized and fairly successful, there are still critical design

and medium problems which hinder the potential of such projects. Anderson, Christensen, and Allen outline BOINC's design goals in their paper [9]: "Volunteers must be given incentives to participate. What constitutes an incentive varies between volunteers. Some enjoy seeing application graphics, either in a window or as a screensaver. Others are motivated by competition with respect to computer speed and donated CPU time; BOINC must therefore make accurate and cheat-resistant measurements of the work done by each host. This data is used by independent web sites that show lists of top users and teams" [9].

1.1 Current Problems with Grid Computing Projects

BOINC and other standalone frameworks have multiple usage problems. These problems range from user activity and persuasion to medium constraints and portability. While the reasons for requiring all these steps are understandable due to the medium that the programs rely on, there is little incentive for users to follow all of those steps.

1.1.1 Activity

Almost every volunteer computing grid relies on client users to sign up, download, and log into a standalone application. Additionally, to not interfere with PC user's computing experience, the grid only runs when the user is not interacting with the computer. The grid loses many PC users who shut off their computer or laptop after use.

1.1.2 Downloading

Additionally, convincing users to download a program is problematic. The fear of viruses, spyware, and malware dissuades many Internet users from downloading software. Even if a PC user does download the program, there is still the risk that their firewall may block the port the program uses anyway.

1.1.3 Registering

Equally difficult is requiring users to register a profile on a website. The increases of spam emails being sent discourage many users from releasing their email to any entity. Most

grid computing applications and frameworks require user registration, potentially deterring thousands of applicable users.

1.1.4 Awareness

The worst and most difficult problem for current grid computing projects is the lack of awareness. There are millions of PCs that do not contribute to any computing grid. Many PC owners, including many in the technology discipline, have never heard of grid computing projects and do not understand their benefits to science.

1.2 The Web Browser Alternative

The medium, as previously mentioned, is the obstruction. Instead of using standalone executables, the web browser provides an alternative platform and solution to the above steps. The idea of using Java applets and the web browser to conduct parallel computing has been around since 1997 [10]. Due to Java's slow initializing time [11] and the lack of applicable users and websites in the World Wide Web[12], the idea has not been developed at any large scale.

2 Introduction

Today, with advancements in web browser speed and new platforms, web applications have the potential to solve the problems of current distributed computing projects and raise awareness about standalone executable applications. The Internet has grown significantly in the past decade. There are over 1.3 billion Internet users [13] accessing hundreds of billions of websites[14] daily. The potential of volunteer grid computing has not been reached as such a relatively low number of the world's Internet users do not participate in any grid.

An Asynchronous JavaScript and XML (AJAX) [15] distributed computing framework prototype, produced in this research project in 2006, proves the possibilities of distributed computing over a web browser. In late 2007, web based distributed computing[16] presented in a conference. AJAX uses JavaScript to communicate to a web server via HTTP. While AJAX

is a satisfactory solution, many users, due to security concerns, disable JavaScript on their browsers; additionally, JavaScript is much slower than standalone applications, and distributed computing algorithms must be divided into smaller subdivisions than downloadable programs. Thus, the volume of users required to produce comparable results to standalone distributed computing applications is significantly higher.

In a speed comparison test on various programming technologies conducted by Mike Lyda on his website OddHammer.com [17], a Java applet performed 10-100 times better than JavaScript. However, due to the slow initializing of Java, JavaScript is a much easier option for webmasters to adapt. An even better solution is the Adobe Flash 9 player. The test shows Flash 9 being not only faster than JavaScript, but comparable to Java.

This thesis will outline the possibility of a distributed computational grid over a web browser using Adobe Flash. Called the Online Community Grid, this research project entails developing a client "widget" that users will run on their web browsers along with developing the server systems. The research also consists of conducting and evaluating various technical and cognitive experiments with standalone applications and the Online Community Grid and eventually publishing the Online Community Grid to websites.

The research project is named the Online Community Grid because it will rely on webmasters to post the application widget on their website and additionally relies on the community members of that website. The Online Community Grid will target blogs and social community websites as they themselves possess an inter-connected network of web users. Virtual communities and social networking sites such as Facebook and MySpace contain loyal visitors who spread stories, videos, photos, and links to other users. In a sense, these users are computational artifacts that process information and determine those to keep and those to bury. With enough development work, the OCG will attempt to establish itself on these virtual communities to help itself spread to the rest of the World Wide Web. Jonathan Bishop discusses virtual communities in a recent paper [18]. His description of community members can clearly be associated with grids: "Some of the online community's members do not participate, people referred to as lurkers, whereas others who have been in the community for a

long time, referred to as elders, participate regularly and support others. Understanding what drives these individuals and how they chose whether or not to participate will lead to online communities that thrive." Grids rely on dedicated, loyal lurkers and elders. In the Online Community Grid model, elders represent contribution to improving or spreading awareness of the system. Those who do not actively participate in the promotion of the grid can be defined as lurkers. Lurkers simply load the application and begin processing code; however, they neither spread recognition nor contribute to improving the system. Thus, the Online Community Grid is in a sense an online community of individuals who dedicate a portion of their processing power for beneficial research.

In regards to this thesis, a challenging problem to overcome is user reliability. Users in the Online Community Grid are trusted to give accurate results based on accurate parameters and instructions; consequently, this model is susceptible to those posting fraudulent data to the grid. A general method for detecting fraudulent data is described by Wenliang Du, Jing Jia, Manish Mangal, and Mummoorthy Murugesan [19], is to build a trust between the user and the server. An honesty ratio can be constructed by requiring new clients to validate previously completed data. Other methods of catching fraudulent data are repeating instructions to different clients across different networks. If a malicious client sends in fake data, a second client on a different network (ideally a different country) processes the same data and responds. Since there will be a discrepancy between the two clients, a third client from a different network can process the same results thereby matching the valid data.

The term Web 2.0, first coined by Tim O'Reilly in 2005 [20], is given to the new generation of blogs, wikis, and social networks emerging from the world wide web. Some social networks, such as Facebook [21], have released APIs [22] that permit web developers to produce applications on the social network itself. Consequently, programmers can code social networking applications that have access to user profiles, friends, and other personal information from the social network. The impact of the Facebook Platform has affected how businesses operate over the web, and many companies have released ad-supported content on Facebook with the intention of spreading their demographics across social networks. With over 64 million active users [23], the Facebook platform provides companies with an unprecedented

user base. According to Facebook, about 15,000 Facebook applications have been developed and about 140 are added each day [23]. Using Facebook or other social networks as a platform for volunteer distributing computing is not only possible, but valuable. In this research project, the Online Community Grid is applied to a Facebook application and the results are evaluated.

In summary, the accessibility, communicable, and technical attributes are the central affordances of the Online Community Grid. Based on previous work in these components, a compilation of these elements can yield a robust, quality system that contributes to the solution of computational problems.

3 Design

The OCG is designed as a widget with GUI components. In order to ease development and also provide a visual component set for OCG application developers to use, the ASwing framework, a robust open source Flash GUI framework based on the Java Swing library, powers the visual interface of the OCG. The OCG visual interface comprises of multiple graphical tabs to allow the user to select options quickly and efficiently.

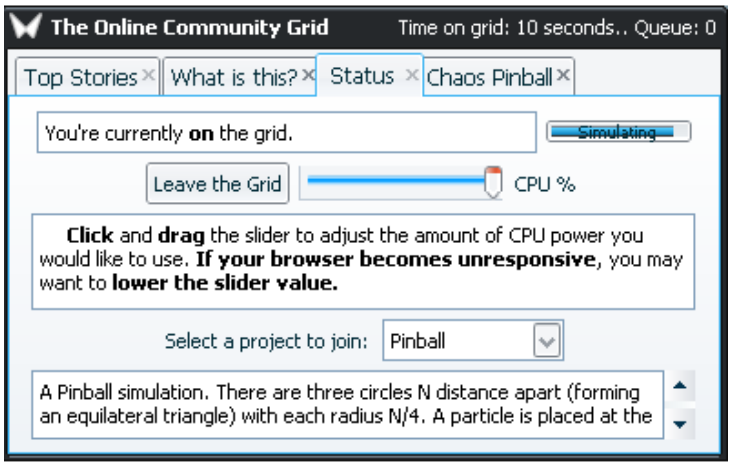

Figure 1: The OCG widget

The OnlineCommunityGrid.com is the official website for this research project. The website features a webmaster and developer section. Developers can find access to

Actionscript optimization techniques as well the API and instructions to create, debug, and release an OCG project. The webmaster section contains instructions on posting the OCG widget to a webpage. A major challenge for the OCG is persuading webmasters to spread the project. In addition to processing computational problems, the OCG's goal is to also provide entertainment and content to website visitors. The current version of the OCG offers syndicated feeds from Yahoo Top Stories. A user can read the news headlines while on the grid by selecting the Top Stories tab. If the OCG simply carried out grid computing processes, webmasters would be less inclined to add the OCG to their website.

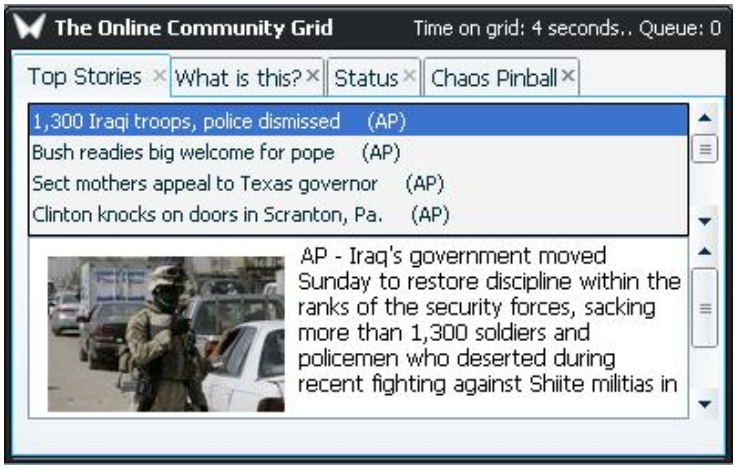

Figure 2: RSS feeder provides content and incentive for webmasters

Since many projects have visualizations and statistics, each project is allotted a GUI tab. In Figure 2, the Chaos Pinball project has its own "project tab" which renders a viewable visualization of the simulation.

3.1 Logistical Design

When a user first loads a webpage with the OCG, the OCG program immediately begins to download. Many Internet users stay on a webpage for only a few seconds; consequently, if the user has downloaded the OCG before, the OCG will load from the cache. The OCG also determines if the user had previously joined a specific OCG project. When a client starts a previously terminated program, grid computing projects prefers resuming rather

than restarting work units. Correspondingly, the OCG has the capacity to resume previous checkpoints of processes.

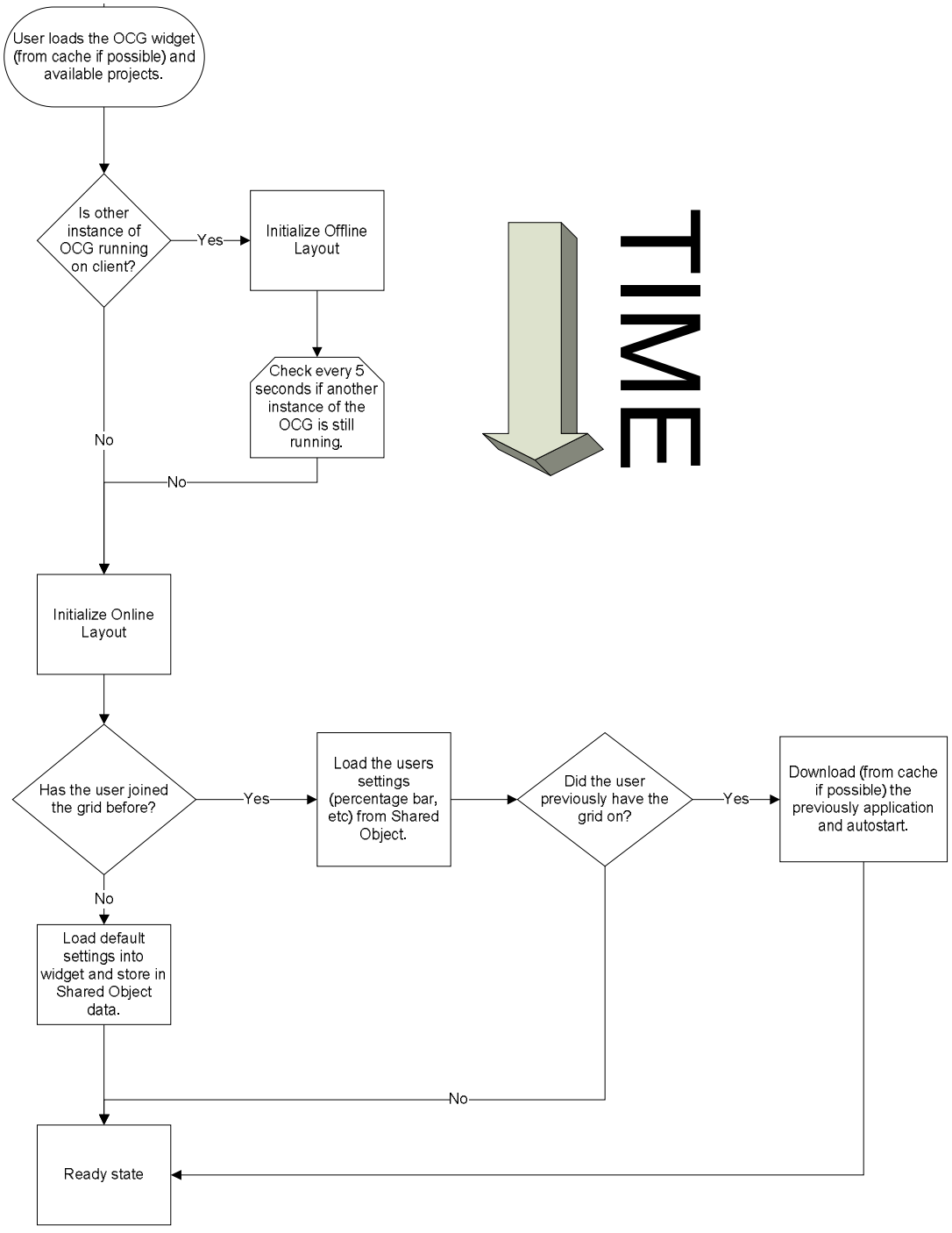

Figure 3: Logistical Flow

3.2 Encapsulation

OCG programmers have access to an API that controls and gathers data from the widget. This API is meant to facilitate development over the OCG. In order to protect the internal structure of the OCG framework, these functions are the only legal means of accessing the framework. The API consists of a variety of functions that include but are not limited to SharedObject (Flash cookie on system) management, ability to access the CPU percentage slider, and HTTP packet queuing.

3.3 Technical Design

The Online Community Grid is a wrapper SWF (Flash) application written in Actionscript 3.0. The application is powered by many open source projects and libraries.

The main framework file is named OnlineCommunityGrid.swf. The backbone of the graphical components of the framework is Aswing. Aswing is an Actionscript 3 port of the Java Swing library. All of the tabs, sliders, textareas, and components of the framework are produced from this library.

To reduce file size the Aswing, xmlSyndication, and corelib libraries are downloaded externally with the FastDLL library. The FastDLL library provides functionality to load external SWF libraries. As shown from Figure 3, when a client loads the framework for the first time, the DLL's are downloaded to the client's cache. Consequently, any modifications of the framework will not result in a new download for every user. This system reduces bandwidth for both the client and server, speeds up initialization time, and ultimately leads to more data processes for applications integrating with the framework.

After the DLL's are loaded to the user's cache, the framework verifies that no other instances are running on the client's machine by attempting to connect to a local network connection using Actionscript 3.0's LocalConnection. If the UDP connection request fails, then the framework is initializes in *offline mode*. Offline mode consists of the basic features of the OCG including RSS feeds and a similar GUI layout provided by Aswing; however, certain tabs

and features are disabled. The framework continuously attempts to connect to the LocalConnection channel every few seconds. When the older instance of the framework is closed, the new instance automatically leaves offline mode and resumes online mode.

Afterwards, the framework instantiates the layout and loads some user saved data. The saved data is comprised of variables such as last CPU slider percentage, last viewed tab, and last viewed project. These saved settings are stored in a Flash Shared Object. The framework then loads a list of official projects from an XML file located on the OCG central server. The list is populated and the last viewed project is selected. If the application was previously running then the framework will immediately load the selected project.

The framework loads the selected project's libraries and code by loading the project's SWF file. After the libraries have been loaded, the SWF file (optionally, but strongly recommended) adds a "Project" JPanel (part of Aswing framework) tab to the Online Community Grid application. The framework then calls a specific function OCG_RUN(API:Object) to the application. This OCG_RUN is the entry point of the project's application and all applications are required to define it in order to function correctly. The API parameter is the OCG_API which is a list of functions that the project can access.

4 Considerations and Obstacles

The OCG must overcome certain medium limitations. The Adobe Flash 9 player, while significantly faster than most browser technologies, is significantly slower than standalone applications written in lower level code. Additionally, most Internet users stay on webpages for short periods of time. Applications utilizing the OCG may be incapable of processing instructions, posting back to the server, or saving checkpoints in very short periods of time. Lastly, there are ethical questions regarding processing data on a client machine without the consent and/or knowledge of the client. The following sections aim to discuss these questions and offer various solutions.

4.1 Rapid Page Browsing

Ideally, users of the OCG would open the OCG as a separate window or, with newer browsers, a separate tab; unfortunately, many users will not. As previously mentioned, many Internet users spend their time browsing web pages quickly. Whenever a user reloads a page with the OCG widget or visits another website, the OCG is terminated. Any work that was processed is automatically deleted from memory. While JavaScript can both detect when the browser leaves the page or the Flash movie closes, its methods are not dependable on every browser. The Online Community Grid application has the capacity to save results, so in order to handle users leaving the page, work done by an OCG integrated application can be saved as a checkpoint LSO file onto the user's machine. These checkpoints can be saved every few seconds and/or upon a dispatch of or JavaScript's window.onbeforeunload. When the application is reloaded, work will resume where previously left off. This functionality and the frequency of checkpoints and uploads is entirely up to the OCG application developer.

4.2 Ethical and Privacy

When a user visits a webpage, he or she's PC immediately begins to process data. The process may be rendering images, loading advertisements, listening to mp3s, or just displaying text. Users are almost always unknowingly downloading data from web servers.

Regardless of that fact, the Online Community Grid application displays on websites as a widget. The application appears as a small rectangle with multiple tabs that provide optional configuration. The two main navigational tabs of the application are a feed tab where a user can read RSS feeds and a Status tab where the user can toggle joining on and off the grid). By default, the Online Community Grid application is turned OFF. However, if the client decides to join the grid, the application will remember the decision and automatically join the next time the application is loaded. The opposite happens when the user chooses to leave the grid. The application will remain off the next time it is loaded.

4.3 Technical

The Adobe Flash 9 platform is powered by the Actionscript Virtual Machine 2.0 (AVM2) [24]. While previous versions of Flash relied on a sluggish virtual machine (AVM), AVM2 significantly improves run-time performance and introduces procedural capabilities which were before thought impossible, impractical, and left for lower level standalone programs. However, despite the increase in performance, Flash 9 applications are significantly slower than lower level C programs and bare a range of technical and security constraints.

These constraints consist of memory access and storage. Current standalone distributed applications are only limited up to the capacity of the machine they run on. Due to security concerns, Adobe Flash rightfully tolerates only up to 100 KB of hard drive space for Adobe Flash by default (much higher than the default 4KB cookie limit on most browsers). The Flash Player will detect when the program attempts to write more than 100 KB onto the client's hard drive. This detection leads to a pop up dialog window requesting more data to be stored. The available storage sizes are 0 KB, 10 KB, 100 KB, 1MB, 10 MB, and Unlimited [25]. Nevertheless, some standalone distributing computing projects require over 10 MB of hard drive to store check points and upon porting to the OCG will require a user permitted increase in storage.

An obvious criticism of the OCG is that due to the rapid speed that user's change webpage, scientific applications will not be able to complete work that sometimes requires hours of processing. A straightforward solution to this argument is to increase the rate of checkpoints to seconds instead of minutes or hours. However, some processor intensive applications will be unable to save checkpoints so frequently, and writing large amounts of data to disk so often is inefficient and can slow down the application.

Fortunately, whenever a Flash SWF is closed, the Flash player automatically flushes the SWF's SharedObjects. As a result, if an OCG application requires a large amount of data to be written to disk, that data can be "queued" to write until the user leaves or reloads a webpage with the OCG.

Furthermore, when running as a web application, Flash shares the same memory thread as the web browser. Consequently, an increase in RAM usage by the Flash will in turn increase the RAM usage of the web browser possibly leading to browser unresponsiveness upon a Flash application requesting or leaking memory. Grid computing applications ported from C/C++ must realize the procedural disparity of Adobe Flash and AVM2.

4.4 Cognitive Affordances

The width and height of the application is small in order to fit with web page designs and varying screen dimensions. Consequently, the OCG widget cannot elaborately explain the intricacies of the framework or thoroughly explain the scientific processes of the integrated applications. To address the small screen display, the OCG uses the ASwing GUI framework and displays the widget with component tabs. Upon the grid connecting successfully to the LocalConnection path, the default tabs are named “Top Stories”, “What is This?”, and “Status”. The “Top Stories” tab shows a list of top news stories from Yahoo! News. “What is This?” explains the purpose and idea of the OCG. The “Status” tab shows which project the client is currently connected with and contains a button to allow the user to join or leave the grid.

5 Results

5.1 Open Source Release

The research project has utilized many open source software and libraries including Eclipse, Filezilla, FastDLL/SWFDLL, ASwing, PHP, MySQL, phpMyAdmin, and Drupal; consequently, the OCG has also been released as open source on Google Code. The code is public to the world but maintained by the OCG team.

5.2 Procedure and Lessons from Integrated Applications

Actionscript 3.0 is slower than lower level languages. Benchmarks conducted in this research show a factor of about 10x-20x difference. The as3Matrix library, an open source library developed in this research project to help catalyze ports of standalone grid applications,

takes about 7 seconds to multiply two 500x500 matrices. Matlab, which utilizes many caching algorithms to optimize Matrix math, takes about 100 ms.

The relative sluggish performance of Actionscript 3.0 is a strong detriment to the OCG. However, due to Flash being a very portable platform, there is a higher potential of high user volume for projects than standalone applications. The OCG also has the ability to market standalone ports through the widget. For example, a BOINC project can be ported to the OCG. The OCG project can then mention to the client the ability to download a more powerful standalone application.

5.2.1 Adobe Alchemy

Adobe released a C/C++ to Actionscript technology named Adobe Alchemy in the labs section of their website in late 2008[26]. The technology converts C/C++ applications to Adobe SWF files. The converted SWF files run up to 10x the speed of native Actionscript code. Scientists can use Alchemy to convert their current projects written for BOINC or other frameworks to Flash more easily.

6 Analysis and Discussion

Rather than a competitor to BOINC or the World Community Grid, the Online Community Grid is more of a catalyst and marketing tool. Grid project developers will not see comparable results between a web browser and a standalone application; however, the portability of the Flash platform provides an advertising avenue and additional computer power for these standalone projects.

7 Works Cited

[1] C. Boulton. (2004, *IBM's Blue Gene Supercomputer is For Sale*. Available: http://www.internetnews.com/ent-news/article.php/3432221

[2] N. Rimon. (2007, *Folding@home Petaflop Barrier Crossed*. Available: http://blog.us.playstation.com/2007/09/25/foldinghome-petaflop-barrier-crossed-update/

[3] G. Woltman. Feb. 10). *The Great Internet Mersenne Prime Search*. Available: http://www.mersenne.org/prime.htm

[4] T. Phillips. (1999, Feb. 10th). *ET, phone SETI@home!* Available: http://science.nasa.gov/newhome/headlines/ast23may99_1.htm

[5] K. A. Douglas*, et al.*, "Spinoff successes of the SETI@home project," *Astrobiology,* vol. 7, pp. 510-510, Jun 2007.

[6] M. Shirts and V. S. Pande, "Computing - Screen savers of the world unite!," *Science,* vol. 290, pp. 1903-1904, Dec 2000.

[7] J. Topolsky. (2007, Feb. 10). *Folding@Home recognized by Guinness World Records*. Available: http://www.engadget.com/2007/10/31/folding-home-recognized-by-guinness-world-records/

[8] D. Graham-Rowe, "BOINC puts idle PCs to work," *New Scientist,* vol. 180, pp. 28-28, Dec-Jan 2003.

[9] D. P. Anderson*, et al.*, "Designing a Runtime System for Volunteer Computing," in *Supercomputing, 2006. SC '06. Proceedings of the ACM/IEEE SC 2006 Conference*, 2006, pp. 33-33.

[10] B. O. Christiansen*, et al.*, "Javelin: Internet-based parallel computing using Java," *Concurrency-Practice and Experience,* vol. 9, pp. 1139-1160, Nov 1997.

[11] J. Neffenger. (1997, Feb. 10). *Results of first-ever JVM server benchmark revealed*. Available: http://www.javaworld.com/javaworld/jw-12-1997/jw-12-volanomark.html

[12] R. H. o. Zakon. (2006, Feb. 10). *Hobbes' Internet Timeline - the definitive ARPAnet & Internet history*. Available: http://www.zakon.org/robert/internet/timeline/

[13] (2007, *World Internet Usage Statistics Top Languages*. Available: http://www.internetworldstats.com/stats7.htm

[14] *March 2008 Web Server Survey*. Available: http://news.netcraft.com/archives/web_server_survey.html

[15] J. J. Garrett. (2005, *Ajax: A New Approach to Web Applications*. Available: http://www.adaptivepath.com/ideas/essays/archives/000385.php

[16] F. Boldrin*, et al.*, "Distributed Computing Through Web Browser," in *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, 2007, pp. 2020-2024.

[17] M. Lyda, "Flash Actionscript performence vs JavaScript," 2006.

[18] J. Bishop, "Increasing participation in online communities: A framework for human-computer interaction," *Computers in Human Behavior,* vol. 23, pp. 1881-1893, Jul

2007.

[19] W. Du*, et al.*, "Uncheatable grid computing," in *Distributed Computing Systems, 2004. Proceedings. 24th International Conference on*, 2004, pp. 4-11.

[20] T. O'Reilly. (2005, *What is Web 2.0*. Available: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[21] M. Zuckerburg. Feb. 10). *Facebook* | *Home*. Available: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[22] M. Zuckerburg, "f8 keynote," ed: Facebook, 2007.

[23] (2008, Feb. 10). *Facebook* | *Statistics*. Available: http://www.facebook.com/press/info.php?statistics

[24] Adobe. (2007, ActionScript Virtual Machine 2 (AVM2) Overview.

[25] Adobe. *Adobe - Flash Player: Settings Manager*. Available: http://www.macromedia.com/support/documentation/en/flashplayer/help/settings_manager.html

[26] Adobe. (2008, 2010). *Adobe Labs - Alchemy*. Available: http://labs.adobe.com/technologies/alchemy/